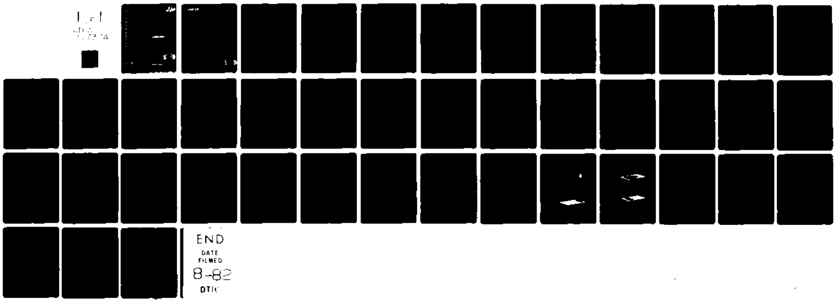
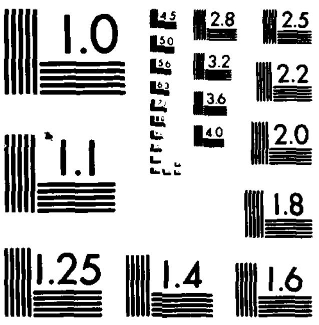
AD-A117 374   JAYCOR ALEXANDRIA VA   F/G 8/3
ELLIPTIC EQUATION SOLVERS FOR NORDA OCEAN MODELS AND A STUDY OF--ETC(U)
JUN 82   A J WALLCRAFT   N00014-81-C-0085
UNCLASSIFIED   J206-82-001/6204   NL



END
DATE
FILMED
8-82
DTIC



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU

JAYCOR

[illegible]

JAYCOR

ELLIPTIC EQUATION SOLVERS FOR NORDA OCEAN MODELS AND A STUDY OF EQUATORIAL OCEAN DYNAMICS

J206-82-001/6204

Alan J. Wallcraft

Prepared for:
Naval Ocean Research and Development Activity
NSTL Station, MS 39529

Under:

Contract Number N00014-81-C-0085

June 2, 1982

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER J206-82-001/6204	2. GOVT ACCESSION NO. AD-A117374	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) ELLIPTIC EQUATION SOLVERS FOR NORDA OCEAN MODELS AND A STUDY OF EQUATORIAL OCEAN DYNAMICS		5. TYPE OF REPORT & PERIOD COVERED Final Report 11/25/80 - 11/24/81
		6. PERFORMING ORG. REPORT NUMBER J206-82-001/6204
7. AUTHOR(s) Dr. Alan J. Wallcraft		8. CONTRACT OR GRANT NUMBER(s) N00014-81-C-0085
9. PERFORMING ORGANIZATION NAME AND ADDRESS JAYCOR 205 South Whiting Street Alexandria, VA 22304		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS A002
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Ocean Research and Development Activity NSTL Station, MS 39529		12. REPORT DATE June 2, 1982
		13. NUMBER OF PAGES 39 pages
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Naval Ocean Research and Development Activity NSTL Station, MS 39529		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

TABLE OF CONTENTS

I. ELLIPTIC SOLVER DEVELOPMENT

TRIDIAGONAL SOLVERS

Several of NORDA's elliptic solvers calculate the solution of a number of tridiagonal systems as part of their total solution process.

If just one tridiagonal system is to be solved on a vector machine then the standard algorithms, variants of Gaussian elimination, are not suitable due to their recursive nature. Probably the best method under these circumstances is scalar cyclic reduction (Swartztrauber, 1979), which has a higher operation count than Gaussian elimination but is vectorizable with average vector length N/Log_2N on systems of dimension N. Even this method is only totally satisfactory for systems of dimension 0(1,000), and hence potentially powerful elliptic solvers requiring the sequential solution of many tridiagonal systems of dimension 0(100) (e.g., block cyclic reduction) are not viable on vector processors. However the tridiagonal systems in NORDA's elliptic solvers can be solved in parallel. Thus Gaussian elimination is vectorizable with a potential vector length at least as large as the number of parallel systems, and on vector machines with only inner loop vectorization (e.g., CRAY-1, Cyber 203/205), it is clearly the best method under these circumstances (Swarztrauber, 1980). However the situation is less clear on the TIASC; suppose the solution of M systems, each of dimension N, is desired on a two pipe TIASC, then:

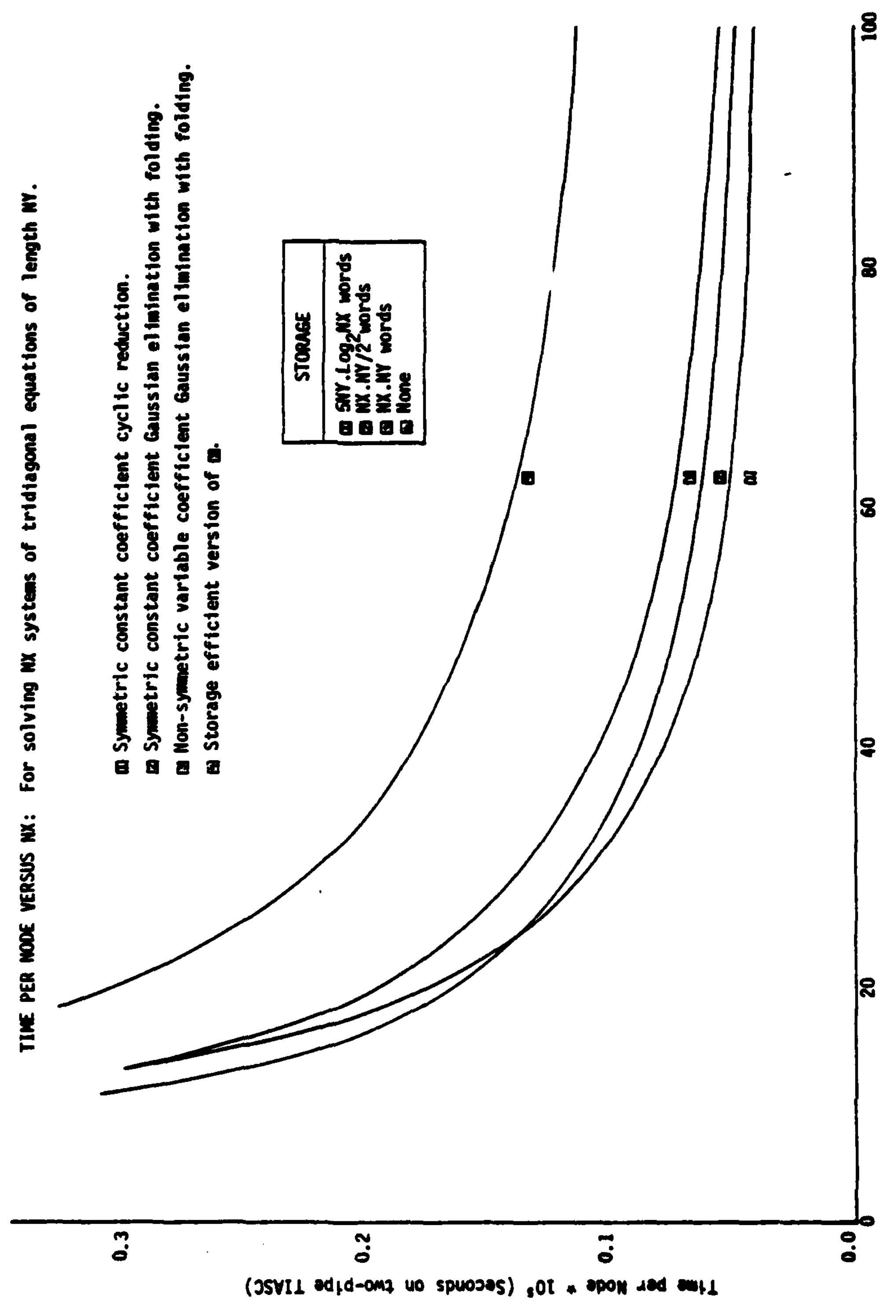
- Standard Gaussian elimination has a vector length of M/2.
- Gaussian elimination with folding has vector length of M.
- Scalar cyclic reduction has a vector length of $M.\ \mathrm{Log}_2N$.

The reason that a 'non standard' version of Gaussian elimination should be used is that the TIASC has two independent vector pipes but standard Gaussian elimination must perform its vector operations in strict sequence. The effect is that each vector operation is shared between the two pipes, giving rise to two vector start-ups per operation (or an average vector length of M/2). The version with folding eliminates off-diagonal elements from the top and bottom of each system working into the center, then backsubstitutes from the center to the top and bottom. It has the same operation count as the standard version but two independent sequences of vector operations (on the top and bottom halves of systems, respectively) can now be performed in parallel on the two pipes, giving rise to a vector length throughout of M. This algorithm has been known for some time; it is used, for example, in the LINPACK package (Dongarra, et al., 1979) to reduce the overheads due to loop control logic on scalar machines, and by others (Temperton, 1980) because on constant coefficient symmetric systems it requires half the storage of the standard version. Despite the fact that it is ideally suited to the TIASC, this is, to our knowledge, the first implementation of the method on this machine. Two classes of tridiagonal systems need to be considered:

- Constant coefficient symmetric systems - for uniform-uniform Helmholtz solvers.
- Variable coefficient non-symmetric systems - for stretched-uniform and stretched-stretched Helmholtz solvers.

In both cases all systems are diagonally dominant so pivoting for stability is unnecessary.

In the constant coefficient case, scalar cyclic reduction (with precomputed coefficients) is faster than all variants of Gaussian elimination: see Graph 1. It usually requires less storage than the latter and it is hence the method of choice for this class of systems. In the variable coefficient case, cyclic reduction requires a prohibitive amount of storage for constants. Moreover, Graph 1 shows that Gaussian elimination with folding using one mesh of storage for constants (i.e., the diagonal factors) is very fast, and it is the method chosen for these systems. Note that a variant using two meshes of storage would be slightly faster (but the difference in time does not offset the increased storage cost) and that a variant requiring no storage space for constants is significantly slower (due mainly to the cost of performing divisions on the TIASC). The latter method is still retained as an option in the stretched-stretched Helmholtz solver



TIME PER NODE VERSUS NX: For solving NX systems of tridiagonal equations of length NY.
$\boxed{1}$ Symmetric constant coefficient cyclic reduction.
$\boxed{2}$ Symmetric constant coefficient Gaussian elimination with folding.
$\boxed{3}$ Non-symmetric variable coefficient Gaussian elimination with folding.
$\boxed{4}$ Storage efficient version of $\boxed{3}$.
STORAGE
$\boxed{1}$ 5NY.Log_2NX words
$\boxed{2}$ NX.NY/2 words
$\boxed{3}$ NX.NY words
$\boxed{4}$ None
0.0
0.1
0.2
0.3
Time per Node * 10^5 (Seconds on two-pipe TIASC)
0
20
40
60
80
100
NX = NY

Graph 1

because the tridiagonial solutions account for only a small percentage of the total solution time in large problems.

RECTANGULAR HELMHOLTZ SOLVERS

NORDA's finite difference Helmholtz solvers can be characterized by four parameters; two describing the boundary conditions on the vertical and horizontal sides, respectively, and two describing the type of mesh spacing (uniform or stretched) in the horizontal, or x, direction and the vertical, or y, direction, respectively. Solution methods are most conveniently grouped according to mesh spacing type and this system is used below. In this report the term 'Neumann boundary conditions' will always apply to the staggered grid Neumann boundary conditions encountered in ocean models used at NORDA.

Uniform-Uniform Solvers

These solvers are all implementations of the Fast Fourier Transform method due to Hockney (Hockney, 1970), which is most commonly denoted as FACR(0). Neumann-Neumann, Dirichlet-Dirichlet, Neumann-periodic and Dirichlet-periodic variants are available. In each case the horizontal dimension is arbitrary but the vertical (internal) dimension is restricted to $2 \times 2^p + q$, or $3 \times 2^p + q$, or $5 \times 2^p + q$ where the largest p is (approximately) 7 and q is -1, 0, or +1 for Dirichlet, Neumann and periodic cases respectively. Solvers with periodic boundary conditions on the vertical sides are not implemented because of the lack of a periodic tridiagonal solver. Probably the only important example of this type is the periodic-periodic solver and this is not currently required at NORDA.

The uniform-uniform solvers are the fastest and most storage efficient available at NORDA and are probably the most heavily used. No significant changes have been made to the TIASC versions of these solvers this year.

Stretched-Uniform Solvers

The uniform-uniform FACR(0) method described above can also be used in stretched-uniform cases. The only required modification to the solver is the replacement of a symmetric constant coefficient tridiagonal solver with a non-symmetric variable coefficient version. The resulting codes are very nearly as fast as the original uniform grid solvers (less than five percent slower) but require more storage. Note that the stretched coordinate is always associated with Neumann or Dirichlet boundary conditions.

The stretched-uniform solvers were originally supplied with a standard Gaussian elimination tridiagonal solver requiring two meshes of storage for constants. This has been replaced by a non-standard version of Gaussian elimination which is faster (on the TIASC) and requires only one mesh of storage for constants. Stretched-uniform solvers should probably see more use at NORDA than they have up to now.

Stretched-Stretched Solvers

The lack of a good stretched-stretched solver was seen as a major shortcoming of NORDA's program collection and two new implementations of very different methods have been produced to fill this gap. The only stretched solver previously available at NORDA was a generalized block cyclic reduction code (Swarztrauber and Sweet, 1975); however, this is not suitable for use on vector machines. The obvious alternative is matrix eigenvalue decomposition (MED) (Farnell,1980), but this has a larger asymptotic operation count and has been found, experimentally, to be slower than cyclic reduction (in FORTRAN on scalar machines) over the range of practical grid sizes.

The major expense in the solution phase of MED is two matrix-matrix multiplications, each with a scalar operation count of 2 NX^2. NY on an NX by NY mesh. But on the TIASC the operation count can be halved, to NX^2.NY, by utilizing this machine's fast dot product capability, and thus almost halving the total solve time. It should be noted that similar relative speeds are obtainable on most vector processors (e.g., TIASC, CRAY-1, Cyber 205) and many scalar machines (by using an optimized machine code routine), but not on the Cyber 203. MED still has an operation count of O(NX^2.NY) compared to O(NX.NY. log NY) for the FACR(0) routine used on stretched-uniform problems. However on the TIASC it is only three times slower at NX = 64, which is acceptable for a stretched-stretched solver (generalized block cyclic reduction is about four times slower than FACR(0) on scalar computers).

Noteworthy features of the MED implementation at NORDA include:

- The required eigensystem is calculated using REAL*8 arithmetic in a preprocessing stage, but may be stored and used in the solution phase in REAL*4 where appropriate.
- NX must be no greater than NY, for efficiency of storage and time. There are no other restrictions on the size of NX or NY.
- Two versions of the required tridiagonal solver are supplied: the faster version requires one mesh of storage for constants, the slower requires no storage and gives rise to just six percent increase in total solve time at NX = 100. The latter version may be preferred when storage is the critical resource.
- Total constant storage requirements are 2 NX^2 or 2 NX^2 + NX. NY words, depending on the tridiagonal solver used.
- The solver has a similar calling sequence to the FACR(0) solvers.

The alternative stretched-stretched solver uses a stabalized marching method. This solver is not usually competitive with MED because it has a very high storage requirement and must be run in REAL*8. If REAL*4 accuracy is sufficient, MED is faster for all practical grid sizes. Marching can be applied to more general problems than Helmholtz's equation on a rectangle and so the method will be discussed more fully in a latter section.

Rectangular Solver Statistics

Storage and time per node statistics are given, in Table 1 and Graph 2, for various Neumann-Neumann solvers on square regions (i.e., NX = NY). A rectangle with NX = 2 NY might be more typical of those encountered in practice but timing information is twice as expensive to obtain in this case. Graph 2 is approximately valid for general rectangular regions provided the smaller of NX and NY is used on the bottom axis for the stretched-stretched solvers and NY is used for the FACR(0) solvers. Table 1 also carries over to rectangles except that MED values (NNSSH.D) represent upper limits, for example if NY = 2 NX

Table 1: RECTANGULAR SOLVER STORAGE REQUIREMENTS (ON TIASC)

NX = NY / SOLVER	STORAGE (REAL*4 MESHES)			
	16	32	64	128
NNUUH.F	1.2	0.8	0.5	0.3
NNSUH.F	1	1	1	1
NNSSH.D	3	3	3	3
	2	2	2	2
MMSSH.M	8	16	32	60
	8	12	20	40

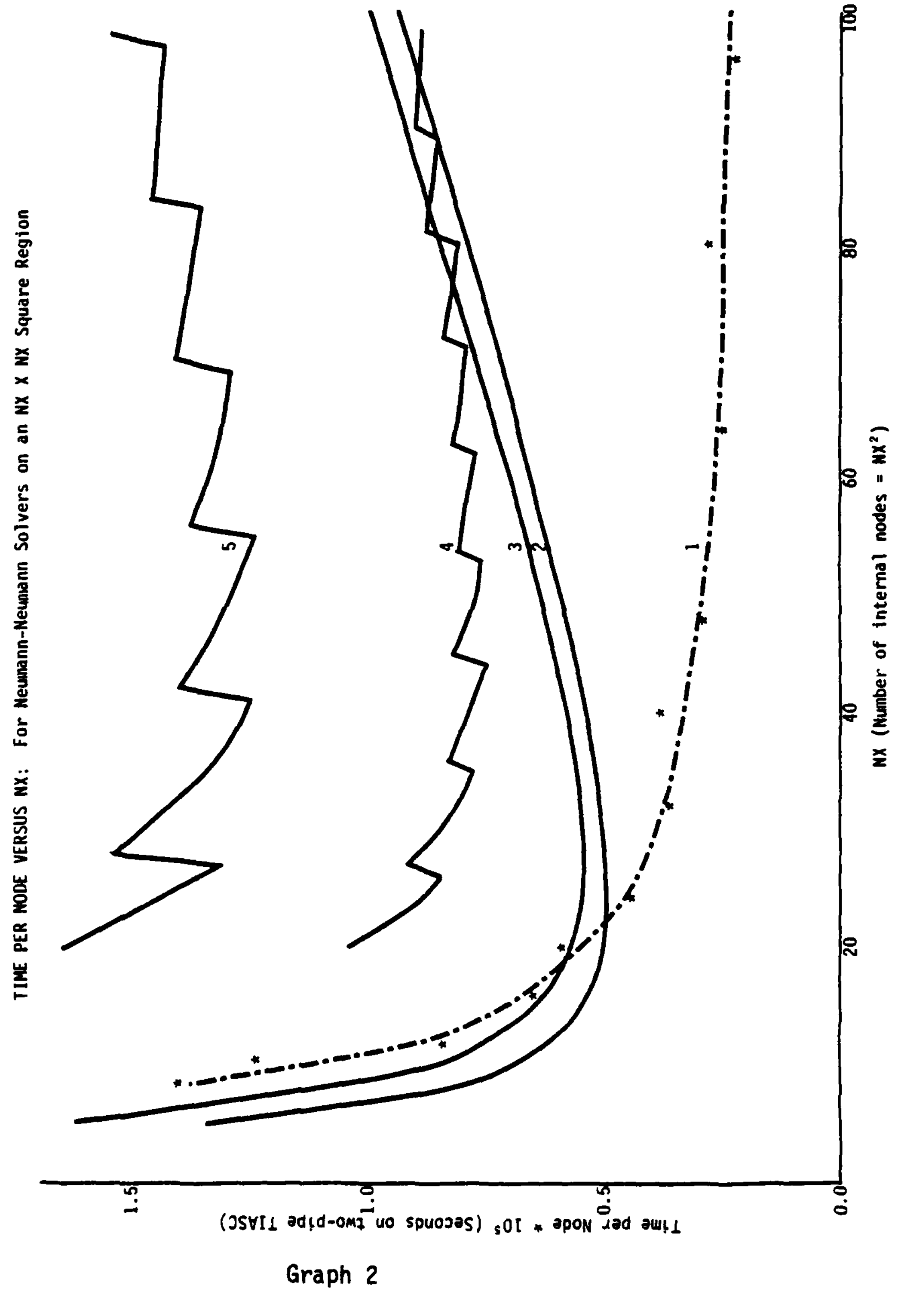
NNUUH.F - a Neumann-Neumann uniform-uniform FACR(0) Helmholtz solver.

NNSUH.F - a Neumann-Neumann stretched-uniform FACR(0) Helmholtz solver.

NNSSH.D - a Neumann-Neumann stretched-stretched MED Helmholtz solver.

MMSSH.M - a mixed-mixed stretched-stretched stabalized marching Helmholtz solver.

Note: Where appropriate the storage requirement of the fastest mode is given above the requirement of the minimum storage mode.



TIME PER NODE VERSUS NX: For Neumann-Neumann Solvers on an NX X NX Square Region
Time per Node * 10⁵ (Seconds on two-pipe TIASC)
NX (Number of internal nodes = NX²)
0.0
0.5
1.0
1.5
20
40
60
80
100
1
2
3
4
5

Graph 2

GRAPH 2

1) NNUUH.F for NX = 8, 10, 12, 16, 20, 24, 32, 40, 48, 64, 80, and 96.

2) NNSSH.D fastest mode.

3) NNSSH.D storage efficient mode

4) MMSSH.M fastest mode (no relaxation sweeps).

5) MMSSH.M storage efficient mode

Note:

a) Times for NNSUH.F (not shown) are within five percent of those for NNUUH.F

b) NNUUH.F, NNSSH.D times are for REAL*4 calculations on the TIASC.

c) MMSSH.M times are for REAL*8 calculations on the TIASC with solution accurate to five significart decimal digits.

d) Preprocessing time is not included, this can be large for NNSSH.D and MMSSH.M.

this solver's storage requirements would be 2 or 1 meshes instead of 3 or 2 meshes respectively.

Storage requirements of all the solvers except the marching code are reasonable, although the amount of storage required increases with the complexity of the grid. The marching solver has a very large storage requirement, for example sixty 128 x 128 meshes is 960 K words and the overhead increases relative to the other solvers with increasing rectangle size. This disadvantage is accentuated on machines such as the TIASC because the marching solver must always run in REAL*8. If double precision accuracy were required in the solution then the storage requirements of all the other solvers would double but that of the marching code would increase only slightly. However martix eigenvalue decomposition still has a significant storage advantage even in REAL*8.

As expected from operation counts the FACR(0) codes are the fastest. But note that the rectangle (actually NY only) is restricted to fixed sizes and that the 5 x 2^p solver is slightly slower than the 3 x 2^p and 4 x 2^p variants. Time per node for these codes decreases with increasing square size, contrary to expectation, because of vectorization effects, which also account for the sharp increase in solve time on small rectangles.

Time per node for the MED solver shows a linear increase with NX as expected from its operation count. It is always faster than the marching solver on practical grid sizes and only takes four times as long as FACR(0) even at NX = 100. Vectorizable effects are less important than with FACR(0) and MED is the fastest solver available at NORDA for very small rectangles. The storage efficient version takes relatively little extra time and may therefore be preferred in some cases.

The graph lines for the marching solver exhibit a characteristic sawtooth shape. This is because the total time includes a component which depends only on the number of sub-blocks used. The time per node jumps whenever the size of the rectangle is such that another block must be added to obtain the required accuracy. The time per node also exhibits a linear trend with NX, but with a far shallower slope than the MED solver. The marching code is certainly fast enough to be a practical stretched-stretched solver, but on rectangular regions MED always requires far less storage and it is also faster for practical grid sizes provided REAL*4 accuracy is sufficient. In storage efficient mode the marching solver requires about two-thirds of the storage as in the standard mode, but takes almost twice as long per solution and is therefore impractical in this context. Marching solvers are not limited to Helmholtz's equation and are almost certainly the fastest known general elliptic PDE solvers. They are also viable Helmholtz solvers for non-rectangular regions, particularly on stretched-stretched grids.

NON-RECTANGULAR ELLIPTIC SOLVERS

Rectangular Helmholtz solvers are the most efficient elliptic PDE solvers available. For this reason, among others, rectangular ocean basins are frequently used at NORDA. However realistic models of actual ocean basins do not give rise to rectangular regions, and so a need exists for non-rectangular Helmholtz solvers. Furthermore one of the layered ocean models occasionally used at NORDA can involve streamfunction equations with a more general elliptic PDE. So a need exists for a non-rectangular general PDE solver. This would also be useful if a non-separable mesh, such as that from a stereographic projection onto the sphere, were to be used in an ocean model.

Non-Rectangular Helmholtz Solvers

Probably the fastest method for this case is the capacitance (or capacity) matrix technique CMT, although the marching methods discussed in the next section can also be competitive, particularly on stretched grids.

The CMT involves solving a related problem in an enclosing rectangle (using say FACR(0)), calculating a correction via a relatively small dense capacitance matrix equation, then solving the corrected rectangular system (using the same rectangular solver). Two variants of this method are available at NORDA:

- The Direct CMT.
- The Preconditioned CMT.

The Direct CMT is the best known variant (Buzbee et al., 1971), it explicitly uses the precalculated and stored capacitance matrix to generate the correction factors. This is very fast but the storage overhead for the capacitance matrix can be very large (between 0 and 25 meshes with 9 meshes being typical). The preconditioned CMT (Wallcraft, 1980), exploits the relationship between the finite difference Helmholtz equation on the rectangle and the capacitance matrix to compute the correction factors iteratively without storing the capacitance matrix. Because each iteration requires a rectangular solve, the already rapid convergence is accelerated by preconditioning with a stored periodic banded approximate inverse to the capacitance matrix. Solution time depends most critically on the size of the Helmholtz coefficient and the required accuracy. With a good initial approximation to the solution, times typically range from slightly faster than the Direct CMT to four times this value. In general the direct variant should be used if sufficient storage is available but the Preconditioned CMT provides a viable alternative, particularly for equations with a large Helmholtz coefficient.

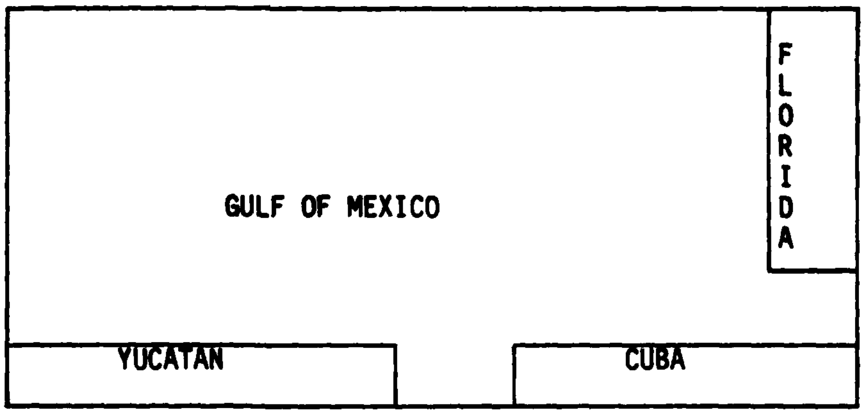
Solvers are available at NORDA for Dirichlet or Neumann problems on uniform grids. Versions for stretched-uniform or stretched-stretched grids could easily be generated. However the latter would use matrix eigenvalue decomposition as a rectangular solver and would probably not be competitive with a marching solver. The package has been written in such a way that it is easy to change from direct to preconditioned variants as required. The latest versions incorporate a much improved user interface and full internal documentation.

Results are presented from a test of the CMT packages on the standard 90 x 50 Gulf of Mexico geometry with tubes at inflow and outflow (Hurlburt and Thompson, 1980) with either a one layer semi-implicit free surface primitive equation model (Neumann boundary conditions) or a one layer quasi-geostrophic model (Dirichlet boundary conditions), together with results from a slightly more realistic geometry for the Gulf with the semi-implicit model only. The times are for 2,000 timesteps each representing 90 minutes model time for a total simulation of 125 days (a practical model simulation would be for about 5 years). All times are in seconds on the two-pipe TIASC.

The storage requirements depend on the number of boundary nodes within the enclosing rectangle. In terms of meshes, the storage requirement of the direct CMT depends only on the region geometry (not on mesh size), and that of the PCMT is typically one mesh or less. The reduced gravity experiment (g = .02) with the semi-implicit model gives rise to a large Helmholtz coefficient and therefore the preconditioned variant is faster than on the barotrophic experiment (g = 9.80). Indeed PCMT is almost as fast as DCMT in reduced gravity cases. The quasi-geostrophic streamfunction calculation involves Poisson's equation and therefore has times very similar to the barotrophic semi-implicit model. But note that the standard iterative Dirichlet CMT (bandwidth = 0) is considerably slower than the preconditioned variant (bandwidth > 0), in contrast with the Neumann case in which the underlying iterative variant is very fast. Solution times for PCMT can vary substantially on different geometries, as the two Neumann examples illustrate, but is relatively insensitive to mesh size, e.g., times for a 10 km grid should be very similar (relative to the DCMT) to those on the 20 km grid shown here. The preprocessing times are larger than might be expected because the matrix inversion routine used has poor vectorization properties. But they illustrate that the PCMT is comparable to the direct variant in the area.

REGIONS

G.O.M. (1) 90 X 50 Mesh



G.O.M. (2) 90 X 50 Mesh

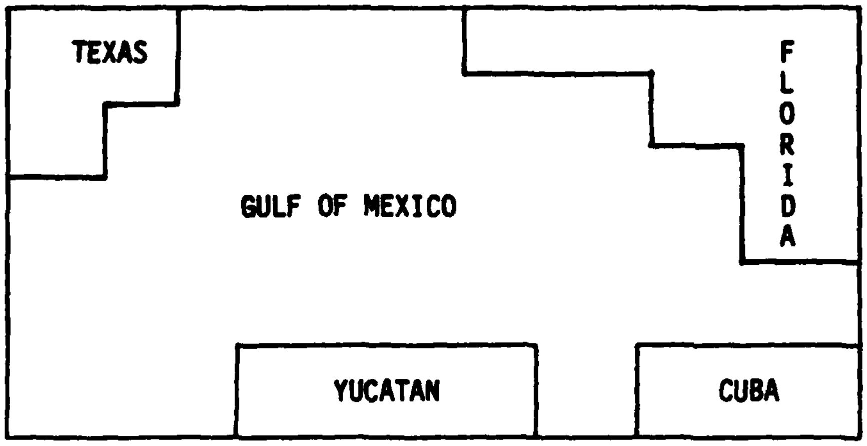


TABLE 2 : G.O.M. (1) - NEUMANN CASE

g	Method	Band-width	Storage (meshes)	Solve time (sec)	Total time (sec)	% of DCMT total	Preprocessing time (sec)
-	DCMT	(131)	4.1	42.8	90.8	100%	14.2
0.02	PCMT	21	0.6	45.8	93.8	103%	11.4
		11	0.3	47.1	95.1	105%	4.7
		∅	0.0	55.8	103.8	114%	1.8
9.80	PCMT	21	0.6	108.5	156.5	172%	11.4
		11	0.3	130.2	178.2	196%	4.7
		∅	0.0	136.6	184.6	203%	1.4

TABLE 3: G.O.M. (1) - DIRICHLET CASE

Method	Band-width	Storage (meshes)	Solve time (sec)	Total time (sec)	% of DCMT total	Preprocessing time (sec)
DCMT	(128)	3.9	40.0	92.2	100%	13.5
	21	0.6	101.5	153.7	167%	9.4
PCMT	11	0.3	112.9	165.1	179%	2.9
	5	0.2	129.3	181.5	197%	1.7
	Ø	0.0	346.6	398.7	433%	0.2

TABLE 4: G.O.M. (2) - NEUMANN CASE

g	Method	Band-width	Storage (meshes)	Solve time (sec)	Total time (sec)	% of DCMT total	Preprocessing time (sec)
-	DCMT	(178)	7.5	45.1	93.1	100%	34.0
0.02	PCMT	27	1.1	48.7	96.7	104%	26.3
		27	1.1	149.1	197.1	212%	25.8
	PCMT	21	0.9	165.4	213.4	230%	14.8
9.80	PCMT	11	0.5	177.1	225.1	242%	5.1
		Ø	0.0	185.7	233.7	251%	1.9

Non-Rectangular General Elliptic PDE Solvers

A variant of the stabilized marching method is the only fully documented and tested solver in this class at NORDA. The algorithm used was originally presented for rectangular regions (Madala, 1978), but NORDA's most recent implementations can handle general bounded regions with Neumann or Dirichlet boundary conditions. The original irregular geometry versions were delivered to NORDA by JAYCOR under ONR Contract Number N00014-80-C-0298 (Ditrich, 1981). The solvers now available at NORDA incorporate minor corrections to the program logic together with more extensive modifications which include:

- Recoding, in FORTRAN, to allow optimal vectorization on the TIASC.
- Replacement of a "bitmap" region representation with a line based representation.
- Merging versions with and without the capability for relaxation sweeps.
- Production of new versions for Helmholtz's equation on stretched-stretched and uniform-uniform meshes.

Stabilized marching algorithms are made up of two sub-algorithms, marching over sub-blocks containing a small number of lines (typically 7-15) and calculating corrections at sub-block boundaries to ensure that the equation is satisfied everywhere. Stabilized marching algorithms differ in how the latter stage is performed, but in this variant the correction phase involves two matrix-vector multiplies at each sub-block boundary. Both the marching and matrix multiply stages are vectorizable but operations applied to block boundaries may only run in scalar mode. This is because the sub-block boundary nodes are of two types, nodes dividing two sub-blocks and nodes within the sub-block but adjacent to the (irregular) region boundary. The natural way to represent such a boundary is to keep a list of the mesh coordinates of each boundary node and it is this representation which gives rise to scalar boundary operations. The vectorizable alternative is to explicitly treat the boundaries in two parts, nodes on the line dividing the sub-blocks are treated as (vectorizable) lines of nodes and nodes in the interior of the sub-block are listed as before. A large fraction of the nodes (perhaps 90 percent) tend to be in the first group allowing significant vectorization of boundary operations. If the region is rectangular there are no nodes in the second group so boundary operations are fully vectorizable. As an example of the difference this modification can make, the original solver takes 145 percent of the time for the fully vectorized code on a typical problem over a 38 by 78 rectangle.

The original code came with different versions for three region types: rectangular, non-rectangular without islands, and non-rectangular with islands. The need for a separate code for rectangles was removed by the above modification to the sub-block boundary data structure. A separate code for cases with islands was still required because a bitmap (actually a REAL array) was used in such cases to prevent the marching process from acting on island nodes. This data structure is again the most obvious but it costs several extra floating point operations per mesh node in each marching phase and an extra mesh of storage. The alternative now used is to list the beginning and end of each strip of sea within each mesh line. The marching process can now be controlled by program logic which only applies it to those nodes on which it is appropriate. The index lists are relatively compact so this data structure saves both storage and time. On a 38 x 78 non-rectangular region with one small island and about 50 percent of mesh nodes on land a code using a bitmap took 120 percent of the time for the current version.

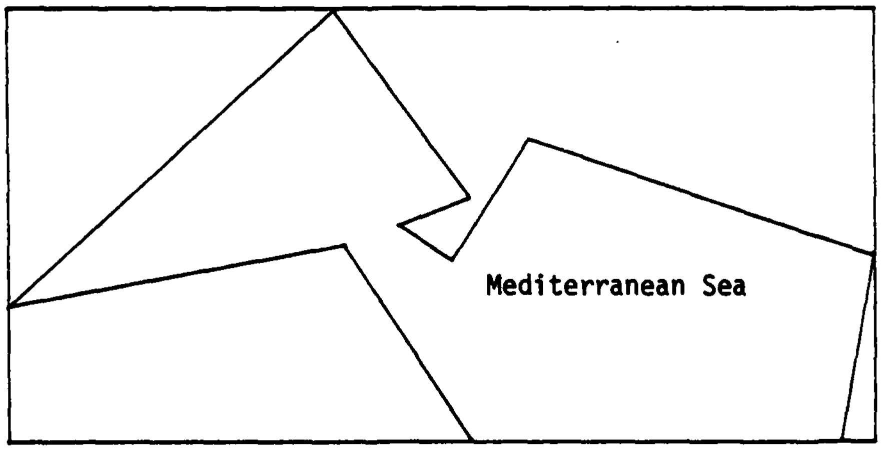
As was shown in Graph 2 marching solvers are very fast and versions have therefore been produced to solve Helmholtz's equation over non-rectangular regions on uniform-uniform or stretched-stretched grids. These versions are required because four of the five mesh arrays needed for PDE coefficients in the general case are not required for Helmholtz's equation. The uniform grid variant is slighlty faster than the general solver, and the stretched grid version slightly slower. There is approximately a 20 percent difference in speed between the stretched solver (which was used in Graph 2) and the uniform code.

Marching solvers and the direct CMT have very different properties, some of which are outlined below.

- The DCMT is a direct method which is generally as accurate as the underlying rectangular solver. The stabilized marching method is also direct but relies on the controlled loss of precision. For this reason it must always run in REAL*8 on IBM style machines (such as the TIASC).
- The storage requirement of the DCMT depends on the region geometry but usually lies between 0 and 25 meshes with 9 meshes being typical. These figures can be halved for Dirichlet problems and in all cases storage requirement (in meshes) is approximately independent of grid length. The storage requirement of the marching solvers is more closely related to region area than to region shape and increases in terms of meshes proportionally to the square root of the number of nodes in the region. Therefore there will be a grid length below which DCMT will require less storage than the marching solvers. However even if the marching code must run in REAL*8 it can require less storage than the direct CMT on some regions with practical grid lengths.
- The solution time of the DCMT depends on the size of the enclosing rectangle and lies between 2 and 3 times that of the corresponding rectangular solver, i.e., FACR(0). Graph 2 indicated that marching solver times lie between 3 and 4 times that of FACR(0) on the TIASC when the latter uses REAL*4 arithmetic. But the marching process is only applied to nodes inside the required region so the rectangular case times must be reduced appropriately. Marching solvers can be faster than the DCMT when a large fraction of the enclosing rectangle lies outside the irregular region. This can occur either because of the intrinsic region shape or because the enclosing rectangle for the CMT is enlarged to conform to the FACR(0) mesh dimension constraints.

Numerical results are presented for non-rectangular solvers on an idealized representation of the Mediterranean Sea for two grid sizes. The equation solved was Poisson's equation with Dirichlet boundary conditions. This region is highly favorable to the marching solver, which is slightly faster than the DCMT in both cases and also requires less storage. The enclosing rectangle used by the CMT solvers is larger than that used by the marching solvers because it must comply with FACR(0) constraints. The storage for the DCMT could be halved in this case since Dirichlet boundaries are involved. The storage requirements of the marching solvers increase faster than those of the DCMT, and the general elliptic PDE version requires significantly more storage than the uniform solver. The PCMT requires relatively little storage but is significantly slower, relative to the DCMT, than in previous examples. This is because the constriction at the ankle of Italy causes the banded preconditioning matrix to be a poor approximation to the inverse of the capacitance matrix.

The uniform Helmholtz marching solver can be competitive with the direct CMT, particularly if REAL*8 accuracy is required, but the latter is probably preferable overall where applicable. The stretched Helmholtz marching code is certainly the fastest available solver for non-rectangular problems, and has no competitors at NORDA. The general PDE solver is very fast but has an even larger storage requirement than the Helmholtz variant. The capability for relaxation sweeps can reduce storage overheads slightly, but a need exists for a more storage efficient solver, comparable in speed and storage requirements to the PCMT for these problems.



IDEALIZED MEDITERRANEAN REGION

TABLE 5: 80 x 40 MEDITERRANEAN REGION SOLVERS

Method	Enclosing Rectangle	Time (sec)	% of DCMT time	Storage (words)
DCMT	81 x 49	.016	100%	30K
MMUUH.M	81 x 41	.014	88%	18K
MMSSG.M	81 x 41	.016	100%	44K
PCMT	81 x 49	.063	394%	8K
PCMT	81 x49	.084	525%	3K

MMUUH.M - uniform grid Helmholtz equation stabilized marching solver.

MMSSG.M - general elliptic PDE stabilized marching solver.

TABLE 6: 160 x 80 MEDITERRANEAN REGION SOLVERS

Method	Enclosing Rectangle	Time	% of DCMT Time	Storage (words)
DCMT	161 x 97	.064	100%	138K
MMUUH.M	161 x 81	.062	97%	114K
MMSSG.M	161 x 81	.071	111%	215K
PCMT	161 x 97	.270	422%	22K
PCMT	161 x 97	.380	594%	14K

SOLVERS ON THE CYBER 203

Access to Fleet Numerical's Cyber 203 has been very limited to date and hence less work than planned has been performed on this machine.

The TIASC and Cyber 203 are both vector processors; that is, full machine speed is obtainable only when performing simple operations on large regularly ordered structures, but they are realizations of vastly different design philosophies. The TIASC is supplied with a very basic set of vector operations and its power lies in the generality of its definition of what constitutes a vectorizable data structure. Almost all (non-recursive, unconditionally executed) floating point or integer operations acting on FORTRAN arrays with subscripts linearly dependent on up to three loop control variables are theoretically equivalent to performing *just one TIASC vector operation.* Furthermore the TIASC's FORTRAN compiler will recognize such operations in standard FORTRAN and generate the appropriate vector code. The Cyber 203, on the other hand, has a very rich set of operations but a very simple definition of vectors, essentially a vector is a one dimensional FORTRAN array.

On paper the Cyber 203 is the more flexible machine, that is any vector operation on the TIASC can be simulated by a sequence of one or more vector operations on the Cyber 203 but there are several (useful) vector operations on the Cyber 203 which are equivalent to scalar code on the TIASC. However some vector operations on the Cyber 203 actually run at scalar, or even slower speed (due apparently to shortcomings in the vector hardware design, which is that of the STAR-100) and these include data motion operations which must be heavily used to compensate for the simplicity of the vector definition. Futhermore the FORTRAN compiler on the Cyber 203 only vectorizes nonrecursive unconditionally executed floating point or integer operations acting on FORTRAN arrays and loop indices which can be trivially transformed into one index acting contiguously on equivalent one dimensional FORTRAN arrays (i.e., it has an inner loop vectorizer with subscript strength reduction). The Cyber 203 is therefore the less flexible machine in standard FORTRAN. The vector hardware problems of the Cyber 203 have been resolved in its successor, the Cyber 205. It is to be hoped that this will motivate CDC to produce a good vectorizing FORTRAN compiler for the Cyber 200 series.

The two-pipe TIASC has a maximum REAL*4 vector speed of 25 M flops and a relatively long vector start up time (half machine speed obtained on vectors of length 90), the Cyber 203 has a maximum mean REAL*8 vector speed of 37 M flops (50 for adds but only 25 M flops for multiplies) and an even larger vector start up time (half machine speed on vectors of length 150), it also has REAL*4 vector capability with maximum speed of 100 M flops but this is not currently available in FORTRAN. The long start up times are not a problem for the explicit portion of three dimensional layered ocean codes since they have an intrinsic vector length of O(10,000) and execute, in FORTRAN, at more than 90 percent of maximum machine speed on either computer. But it is a major factor in elliptic PDE solvers, which tend to operate on the two dimensional mesh line by line, i.e., act on vectors of length O(100). This problem is more serious on the Cyber 203 for two reasons:

- It has the longer vector start up times
- Its FORTRAN vectorizer is significantly less sophisticated than that of the TIASC.

Only one solver (the Neumann-Neumann uniform-uniform Helmholtz solver) has so far been transferred to the Cyber 203 and the work involved illustrates clearly the problems with the FORTRAN compiler. On the TIASC the solver has the form:

- Forward fast Fourier transform (FFT) and transpose.
- Tridiagonal solution by cyclic reduction.
- Transpose and reverse FFT.

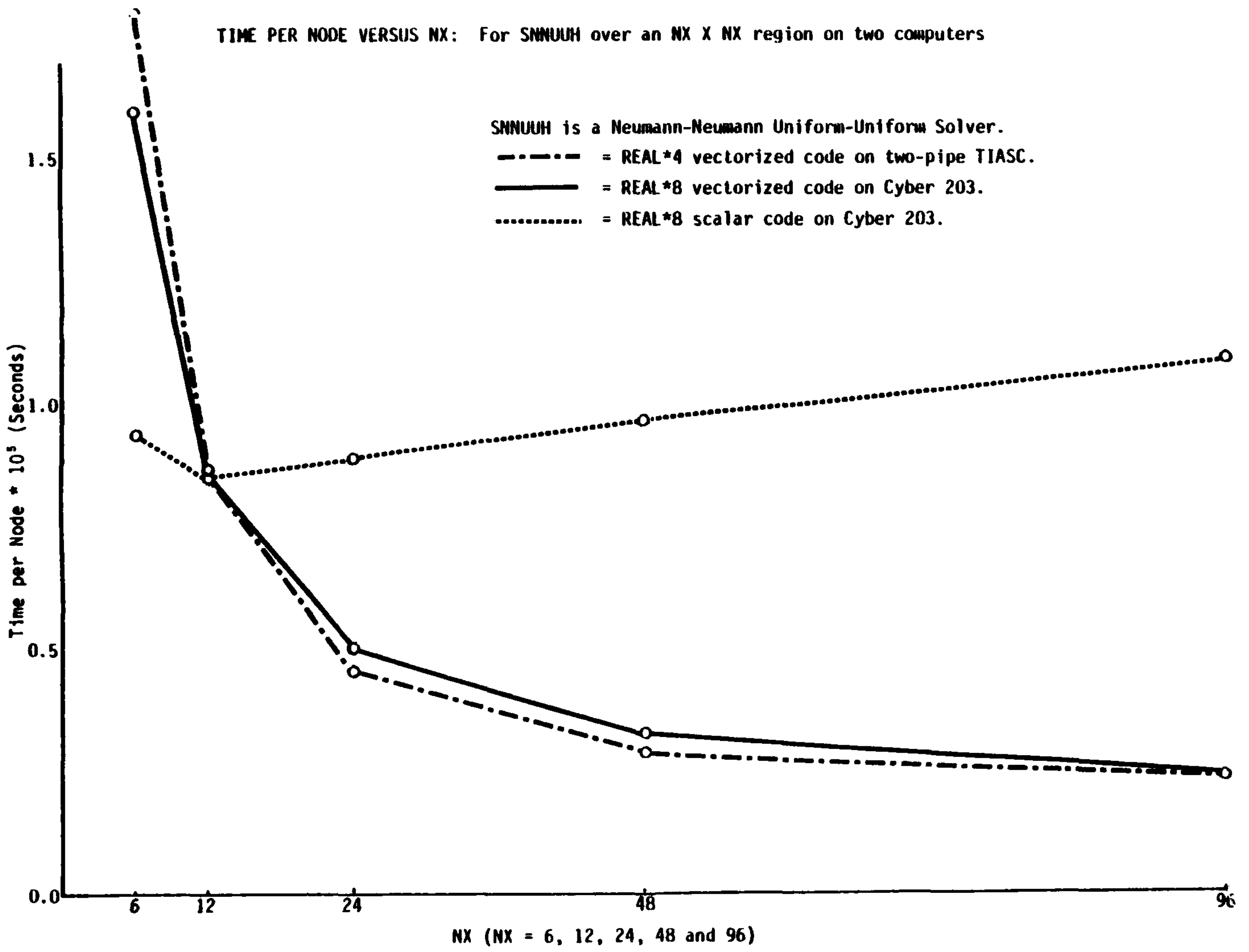
Suppose the grid contains NX by NY internal mesh nodes. On the TIASC the NX forward and reverse FFTs are performed in parallel and each individual FFT is vectorizable with average vector length $O(NY/\log_2 NY)$. Therefore the FFT routines (written in FORTRAN) are vectorizable with average vector length $O(NX.NY/\log_2 NY)$. The FFT routine includes a mesh transpose as part of the unscrambling or scrambling phase to facilitate optimal vectorization of the tridiagonal solution phase. The NY tridiagonal solutions are computed in parallel using cyclic reduction, which has an average vector length of $NX/\log_2 NX$ on individual systems, for a total vector length of NY.NX/$\log_2$ NX. Thus in all phases of the algorithm, vector lengths significantly longer than the NX or NY, indicated by parallelism, are obtained. This is a direct result of the TIASC FORTRAN capability for vectorizing nested loops.

On the Cyber 203 the modified solver has the form:

- Forward FFT
- Transpose
- Tridiagonal solution by Gaussian elimination
- Transpose
- Reverse FFT

With a few minor exceptions an individual FFT is not vectorizable on the Cyber 203 (because the potential vector elements are not contiguous in memory) and hence the FFT phases have average vector lengths of about NX. The mesh transposes can no longer be overlayed with the unscrambling process in the FFT because the FORTRAN compiler does not recognize the corresponding code as vectorizable. A separate routine containing both the FORTRAN code (for use on other machines) and a call to a machine language routine which uses the Cyber 203's fast transpose capability has been introduced. A single cyclic reduction solution is not vectorizable so the tridiagonal solver used is Gaussian elimination vectorized across systems for an average vector length of NY. This method has a lower operation count than cyclic reduction (which was only faster on the TIASC because of its longer vectors) but has a storage requirement of about half a mesh for constants and this is perhaps twice that of the TIASC routine. Because of deficiencies in the Cyber 203 FORTRAN vectorizer the solver can not be set up as a library. It must be recompiled for each new size with dimensions supplied via parameter statements. This is not a serious problem in an ocean modeling group since mesh sizes change relatively infrequently.

Graph 3 shows that the total solution times are very similar on the TIASC and Cyber 203, although the Cyber 203 results are for REAL*8 calculations against REAL*4 on the TIASC. The maximum Cyber 203 REAL*8 speed is about one and a half times that of the TIASC in REAL*4 mode, so the solver is relatively more efficient on the TIASC as expected from the longer average vector lengths on this machine. The fastest Cyber 203 times in scalar mode are also included on Graph 3, illustrating this


TIME PER NODE VERSUS NX: For SNNUUH over an NX X NX region on two computers
SNNUUH is a Neumann-Neumann Uniform-Uniform Solver.
= REAL*4 vectorized code on two-pipe TIASC.
= REAL*8 vectorized code on Cyber 203.
= REAL*8 scalar code on Cyber 203.
Time per Node * 10⁵ (Seconds)
1.5
1.0
0.5
0.0
6
12
24
48
96
NX (NX = 6, 12, 24, 48 and 96)

Graph 3

machine's great speed as a scalar processor. The TIASC is a relatively poor scalar processor and its corresponding scalar times would not lie on this graph.

The major effort involved in transferring the solver to the Cyber 203 does not lie in the modifications, indicated above, required for best possible vectorization, but in recoding necessitated by a bug in the FORTRAN compiler which caused incorrect object code to be generated for some of the FFT routines. The errors are probably due to the complicated subscripting used in these routines (which are written in totally standard ANSI 1966 FORTRAN) and it is therefore likely that they will recur in all the other uniform-uniform solvers. However it will be simpler to correct the remaining eight sets of FFT routines since the nature of the error is not understood. It was the detection and correction (by experimentation) of the errors which was most time consuming in this first set of routines.

II. OCEAN MODEL DEVELOPMENT

NORDA has four types of layered three dimensional ocean models:

- Quasi-geostrophic
- Rigid-Lid
- Explicit free surface
- Semi-implicit free surface.

The semi-implicit free surface codes are the most frequently used at NORDA and they are also the most general, with the capability of handling non-rectangular geometry, prescribed inflow and outflow with self determining profile. The explicit free surface codes have similar capabilities but are limited by constraints on maximum time step to a small class of problems. The rigid-lid and qausi-geostrophic models could only (previously) handle wind driven flows in closed rectangular flat bottomed basins. For this reason (among others) they have found little use at NORDA, although this qausi-geostrophic model is probably the most widely used layer type model in the numerical ocean modeling community outside NORDA.

Versions of the following models have been set up for the standard irregular geometry Gulf of Mexico model region with tubes at inflow and outflow (Hurlburt and Thompson, 1980).

- 2-layer rigid-lid with flat bottom.
- 2-layer rigid-lid with bottom topography.
- 1-layer qausi-geostrophic.
- 2-layer qausi-geostrophic with bottom topography.

Two versions of the rigid-lid model are required because the streamfunction equation involves an elliptic equation more general than the usual Helmholtz equation when bottom topography is introduced. The capacitance matrix technique is used to solve for the streamfunction in the flat bottomed model, but a marching solver must be used in cases with bottom topography. The two layer quasi-geostrophic model allows for bottom topography, but this must be small scale if the model is to be consistant. The rigid-lid inflow and outflow boundary conditions are very similar to those used by the free surface models. In the quasi-geostrophic models the profile of both the inflow and the outflow is prescribed; this is acceptable in this case because the outflow port is narrow.

The models have been set up for the Gulf of Mexico model region but it is relatively easy to change to any other geometry. The quasi-geostrophic models are less flexible in this regard than either the free surface or the rigid-lid models because they currently require the outflow profile to be predefined.

The one layer semi-implicit model has been successfully transferred onto the Cyber 203. A report has been prepared outlining the steps to take when transferring any model onto the Cyber 203 if full two dimensional vectorization is required (Wallcraft, 1981).

III. EQUATORIAL DYNAMICS

Introduction

The theory of solitary waves and their application to geophysics has been the focus of a variety of recent studies. The works of Clarke (1971) on midlatitude Rossby solitons, Maxworthy and Redekopp (1976) on Jupiter's Great Red Spot and Boyd (1980) on equatorial Rossby solitons are just a few examples. For extensive reviews of the literature, the reader is referred to Scott et al., (1973) and Miles (1980).

Boyd (1980) derived analytical expressions for Equatorial Rossby solitons (ERS) using as his governing system the nonlinear form of the shallow water wave equations for a homogeneous layer of fluid. His model ocean was horizontally unbounded. The effects on Rossby solitons of other equatorial waves as well as the mean circulation were neglected. The system was reduced to the solution of the one-dimensional KDV equation with an initial distribution of the height field as the forcing.

The general solution to this governing system, i.e., the "prototype" equatorial Rossby soliton, propagated across the basin with no change in shape or amplitude. In the corresponding linear solution, the same initial condition rapidly evolved into a wavetrain. Boyd found that ERS results from a variety of initial conditions; they are the only stable solution to the initial value problem. He speculated that these solitary waves would most likely be generated at the eastern boundary following a sudden relaxation of the equatorial trades such as that associated with El Nino.

Although Boyd's simple model is a valuable first step, it is important to go beyond the initial value experiments. In this study, a 1½ layer reduced-gravity model of the tropical Pacific is used to examine the generating mechanisms for equatorial solitons. In the first set of experiments, the model is initialized with the prototype soliton of Boyd in order to demonstrate the behavior of the "pure" equatorial Rossby Solitary wave. In subsequent experiments, forcing is applied as a patch of zonal wind stress. It is demonstrated that ERS can be generated by wind events in which the equatorial trades relax for periods of a few weeks to months. The numerical calculations also support Boyd's hypothesis that equatorial solitons may be excited during the El Nino event.

Initial Value Experiments

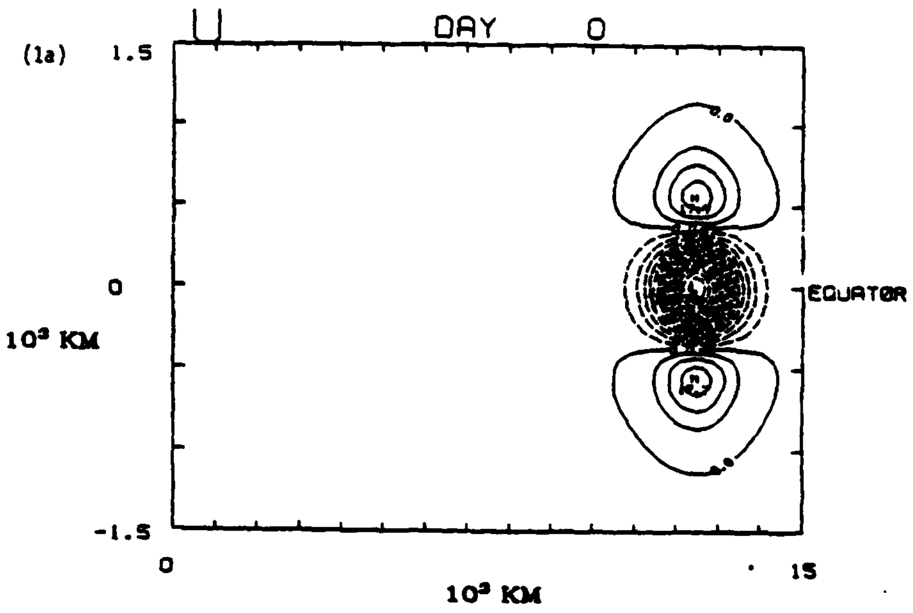
The numerical model is a nonlinear, 1½ layer reduced-gravity model of the equatorial Pacific. The northern and southern boundaries, which are open, are located 1,500 km from the equator. The basin is rectangular in shape and extends 15,000 km zonally. For the experiments described below, the initial upper layer thickness is 150 m and the stratification, $\Delta\rho$, is $3\sigma_t$ units. A more detailed description of the model can be found in Kindle (1979, 1981).

The model is initialized with Boyd's solution for the first latitudinal mode equatorial Rossby soliton (Fig. 1). Case 1 is the linear solution. The effects of dispersion cause the initial feature to evolve into a wavetrain (Fig. 2a). In Case 2 the nonlinear terms are included; Fig. 2b reveals that the initial disturbance propagates across the basin with very little change in shape or amplitude. Clearly, the nonlinear terms balance the tendency to disperse thus demonstrating that the model reproduces the behavior of the equatorial Rossby soliton derived analytically by Boyd.

Wind Generation of Equatorial Solitons

In the experiments described below, wind stress forcing is applied to the model ocean in order to test whether solitons can be excited by wind events. The forcing is in the form of a patch of zonal wind stress with a uniform meridional distribution and a top-hat zonal shape. The wind is directed from west to east and represents a relaxation of the mean equatorial trades. As described by McCreary (1977) and Kindle (1979), the wind patch radiates packets of Kelvin and Rossby waves. The Kelvin waves, which have a much larger amplitude than the Rossby waves, propagate a downwelling pulse toward the eastern boundary. The reflection of this downwelling pulse initiates westward

(1a)



(1b)

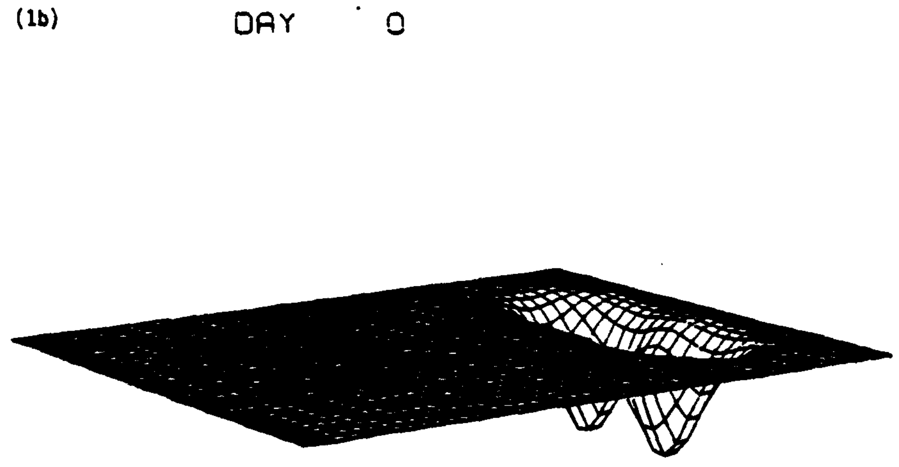
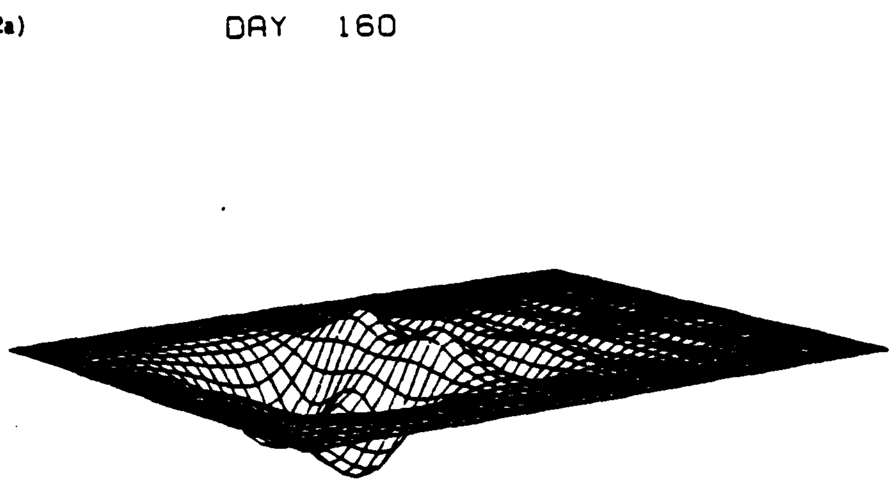


Figure 1: Initial condition for Cases 1 and 2 - the prototype Rossby soliton derived by Boyd. (a) Zonal velocity component. The contour interval is 5 cm/sec. (b) Model pycnocline surface as viewed from the southwest Pacific. The maximum amplitude of the initial depression is 25 meters.

(2a)



(2b)

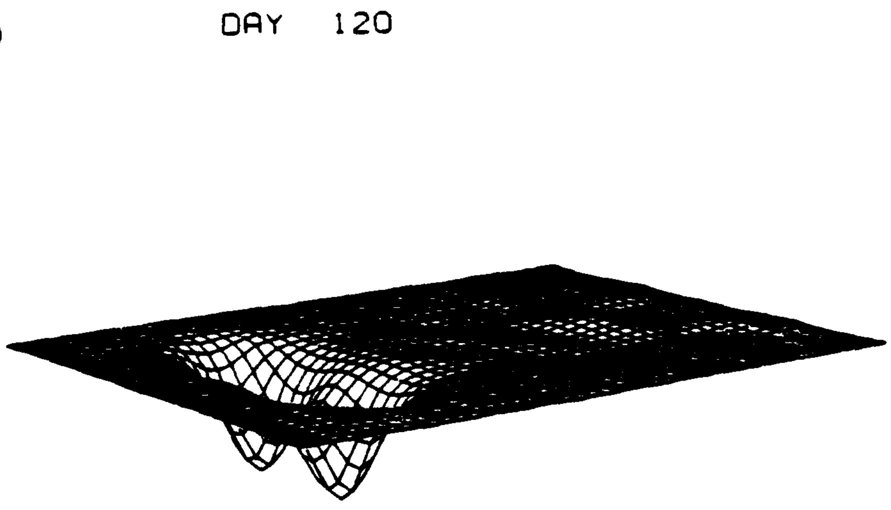


Figure 2: (a) Model pycnocline surface for Case 1 (linear solution). (b) Case 2 (nonlinear solution).

propagating Rossby waves. The subsequent description focuses on the response of the first latitudinal mode Rossby wave created by the eastern boundary reflection.

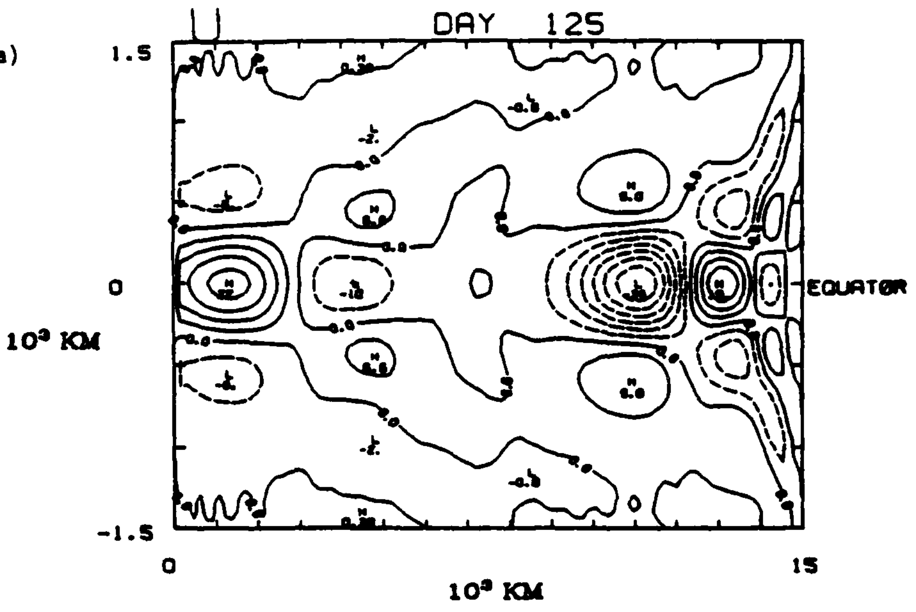
In Case 3, the wind stress has a zonal width scale of 2,000 km and an amplitude of .5 dynes/cm^2. The eastern edge of the patch is located 6,500 km from the eastern boundary. The forcing is applied impulsively and switched off at Day 30. Fig. 3 presents the results of the linear calculation. As the first mode Rossby wave propagates westward, it evolves into a wavetrain. Although not as pronounced as in Case 1, the effects of dispersion are very evident. The nonlinear counterpart to this experiment (Case 4) is shown in Fig. 4. The first mode Rossby wave propagates across the basin and exhibits only slight changes in its shape and amplitude. The differences between the linear and nonlinear experiments and the similarity between this solitary wave and the prototype soliton lead to the conclusion that this wind event has generated an equatorial soliton.

Although not shown, it was found that wind events with time scales from two weeks to one month generate solitons similar to Case 4. This is true even if the amplitude of the forcing is only .25 dynes/cm^2. Events with time scales greater than one month, however, can generate multiple Rossby solitons, the number of which depends upon the product of the time scale, zonal width scale and amplitude of the wind event. Simple El Nino simulations suggest that this intense event would very likely excite multiple solitons subsequent to the reflection of the Kelvin wave front from the eastern boundary.

Summary

Numerical experiments were performed in order to examine the generating mechanisms for equatorial Rossby solitons. It was shown that wind events representing a relaxation of the equatorial trades can generate these solitary waves subsequent to the reflection of the Kelvin wave from the eastern boundary. If the duration of the event is greater than one month, it is possible for multiple solitons to form. This is particularly true for large scale events like El Nino. The numerical calculations support the hypothesis of Boyd (1980) that solitons would

(3a)



(3b)

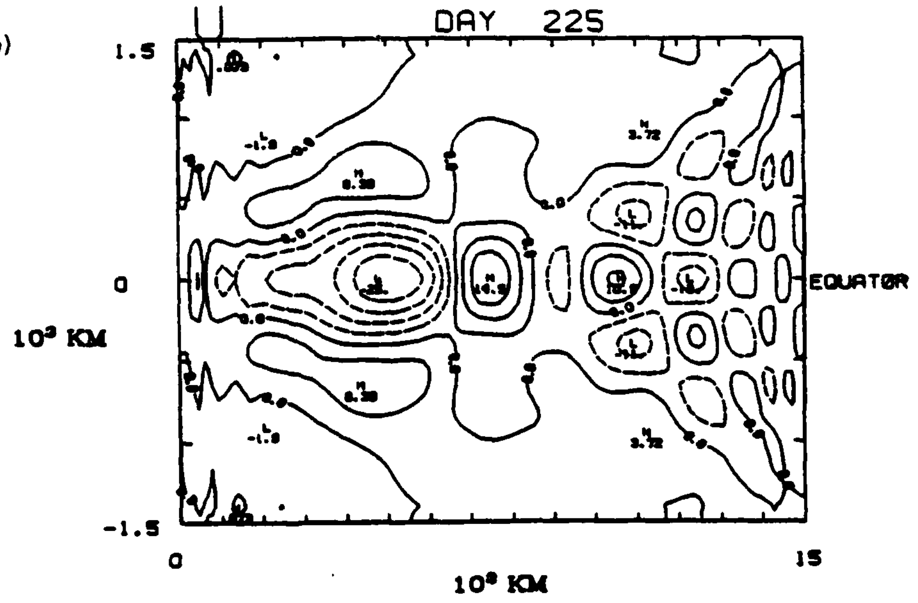
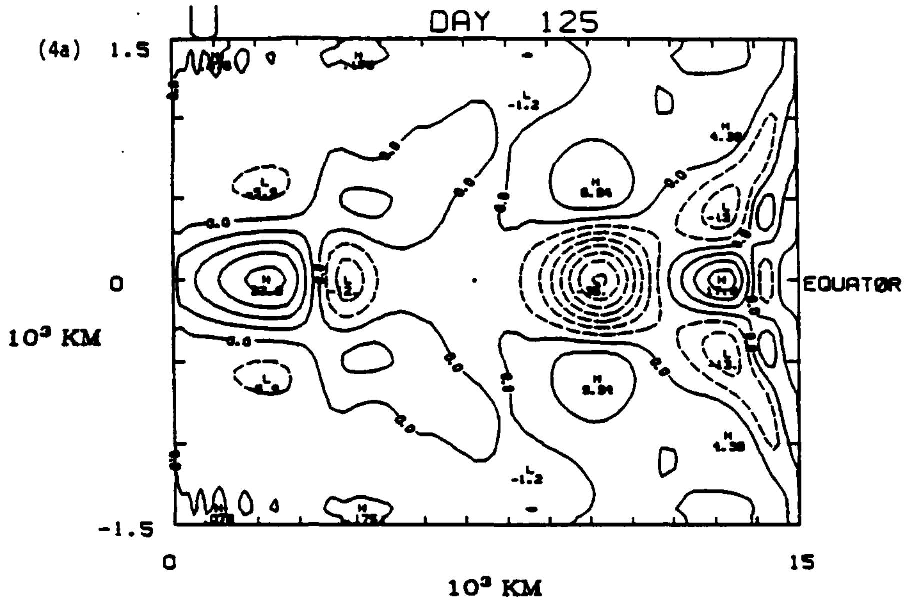


Figure 3: (a) The zonal velocity component at Day 125 for Case 3 (linear solution). The first mode Rossby wave resulting from the Kelvin wave reflection is located 4000 km from the eastern boundary. The contour interval is 5 cm/sec. (b) Same as (a) except at Day 225.

(4a)



(4b)

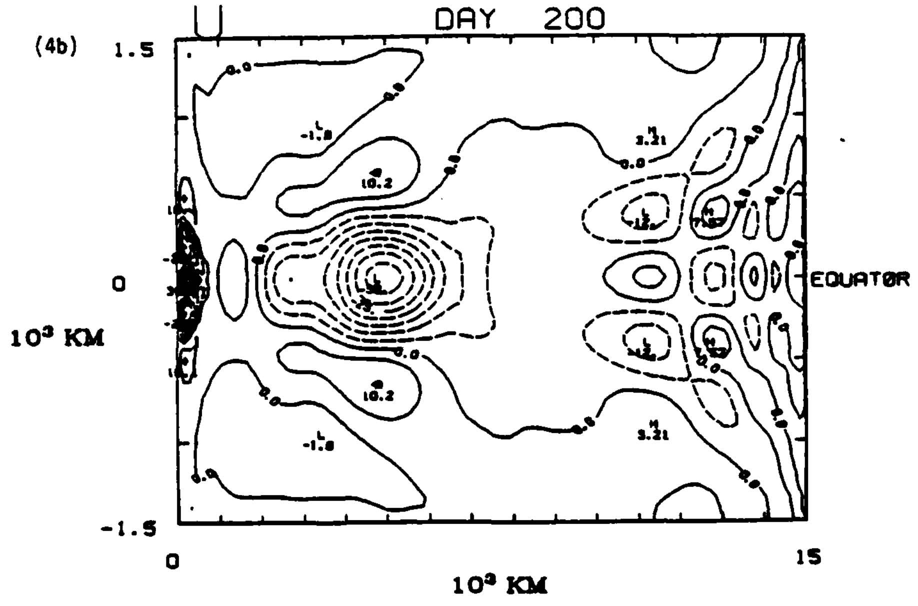


Figure 4: (a) Same as Fig. 3 except for Case 4 (nonlinear solution). (b) Zonal velocity at Day 200 for Case 4. Equatorial soliton is clearly evident at x = 5000 km.

most likely be generated during El Nino events. These results also suggest that even wind events on a smaller scale, both temporally and spatially, can generate Rossby solitons and that these waves may have a significant influence on the variability of the equatorial thermocline.

REFERENCES

Boyd, J.P., 1980: Equatorial Solitary Waves. Part 1: Rossby Solitons, J. Phys., 10, 1-11.

Buzbee, B.L., Dorr, F.W., George, J.A. and Golub, G.H., 1971: The Direct Solution of the Discrete Poisson Equation on Irregular Regions. SIAM J. Numer. Anal. 8 pp. 722-736.

Clarke, R.A., 1971: Solitary and Conidal Planetary Waves. Geophys. Fluid. Dyn., 2, 343-354.

Dietrich, D.E., 1981: A Program of Elliptic Solver Development and Implementation in Semi-implicit Numerical Ocean Circulation Model. JAYCOR Final Report J510-81-053/2192.

Dongarra, J.J., Noler, C.B., Bunch, J.R. and Stewart, G.W. 1979: LINPACK Users Guide. SIAM Publication.

Farnell, L., 1980: Solution of Poisson's Equations on a Nonuniform Grid. J. Comp Phys. 35 pp. 408-425.

Hockney, R.W. 1970: The Potential Calculation and Some Applications. Meth. Comp. Phys. 9 pp. 135-211.

Hurlburt, H.E., Thompson, J.D., 1980: A Numerical Study of Loop Current Intrusions and Eddy Shedding. J. Phys. Oceanogr. 10 pp. 1611-1615.

Infotech Ltd., 1979: Supercomputers. Infotech State of the Art Report.

Kindle, J.C., 1979: Equatorial Pacific Ocean Variability-Seasonal and El Nino Time Scales. Ph.D. Dissertation. The Department of Oceanography. The Florida State University, Tallahassee, FL. 134 pp.

Kindle, J.C., 1981: A Numerical Study of Equatorial Rossby Solitons. Submitted for publication.

Madala, R.V., 1978: An Efficient Solver for Separable and Non-separable Elliptic Equations. Mon. Weath. Rev. 106 pp. 1735-1741.

Maxworthy, T. and L.G. Redekopp, 1976: A Solitary Wave Theory of the Great Red Spot and Other Observed Features in the Jovian Atmoshpere. Icarus, 29, 261-271.

Miles, J.W., 1980: Solitary Waves. Annual Review of Fluid Mechanisms, Vol. 12, Annual Reviews, Inc., 11-43.

Scott, A., et al., 1973: The Soliton: A New Concept in Applied Science. Proc. IEEE, 61, 1443-1483.

Swarztrauber, P.N., Sweet, R., 1975: Efficient FORTRAN Subprograms for the Solution of Elliptic PDEs. NCAR Tech. Note TN/1A-109.

Swarztrauber, P.N., 1979: The Solution of Tridiagonal Systems on the CRAY-1, (Infotech 1979).

Temperton, C., 1980: On the FACR(0) Algroithm for the Discrete Poisson Equation. J.Comp. Phys. 34 pp. 314-329.

Wallcraft, A.J., 1980: The Preconditioned Capacity Matrix Technique. Ph.D. Thesis, Imperial College, U. of London.

Wallcraft, A.J., 1981: Transferring Ocean Models from the TIASC to the Cyber 203. JAYCOR Report J206-81-0161/6204.